

i Om eksamen

Eksamensinformasjon – digital skoleeksamen

Fakultet: Teknologi, kunst og design

Utdanning: Teknologiske fag

Emnenavn: Algoritmer og Datastrukturer

Emnekode: (ORD) DATS2300 / ITPE2300

Dato: 05.12.2019

Tid: 09.00 - 12.00

Antall oppgaver: 5

Tillatte hjelpemidler: Ingen

Merknad:

Råd og tips:

1. Les gjennom hele oppgavesettet før du begynner og planlegg tiden.
2. Svar utfyllende på oppgavene så du viser at du har forstått pensum.
3. Bruk ikke for lang tid på et punkt. Gå isteden videre til neste punkt og eventuelt tilbake hvis du får god tid.
4. Hvis du trenger en hjelpestruktur (liste, stakk, kø o.l.) fra java.util eller fra kompendiet, kan du fritt bruke den uten å måtte kode den selv. Men den må brukes på en korrekt måte og du bør si fra om dette i en kommentar.
5. Hvis du har idéer om hvordan ting skal løses, men likevel ikke klarer å få det til, kan du demonstrere idéene dine med ord, tegninger o.l.

Vurdering:

1. Ved sensurering blir det lagt vekt på hva du viser av forståelse av kursets pensum opp mot læringsutbyttet. Det vil si at det vektlegges hvordan du kommer frem til et svar ved å bruke pensum.
2. De fem oppgavene teller likt.

Lykke til!

1 **Oppgave 1: Køer og stacker**

Denne oppgaven handler om forskjellige typer køer

- a. Beskriv kort hva de forskjellige følgende køene er:
 - i. Kø
 - ii. FIFO-kø
 - iii. LIFO-kø
 - iv. Stack
 - v. Deque
 - vi. Prioritetskø
- b. Se på kildekoden i vedlegget. Beskriv kort hva de fire delene av kildekoden gjør
- c. Hva blir utskrift av kildekoden i vedlegget?

Skriv ditt svar her...

Format

B


I


U


\times_2


\times^2


I_x




















Ω





Σ



Words: 0

Maks poeng: 10

2/6

² Oppgave 2: Quicksort

I denne oppgaven skal du sortere et sett med verdier med Quicksort. I denne oppgaven bruker vi arrayet

```
char[] values = {'B', 'C', 'K', 'A', 'F', 'L', 'T'};
```

- a. Ta utgangspunkt arrayet:
 - i. Forklar hva partisjonering er.
 - ii. Partisjoner arrayet over ved å bruke 'K' som skilleverdi.
 - iii. Hvilken indeks vil skilleverdien ligge på etter partisjonering?
- b. Forklar hvordan quicksort fungerer
 - i. Beskriv, med ord, hvordan quicksort fungerer.
 - ii. Lag en tegning som stegvis viser hvordan quicksort sorterer arrayet over.
- c. Algoritmeanalyse av quicksort
 - i. Vil du bruke iterasjon eller rekursjon for å implementere quicksort?
 - ii. Hva slags kompleksitet har quicksort i gjennomsnittstilfellet? Forklar kort hvordan du kommer frem til svaret ditt.

Skriv ditt svar her...

Maks poeng: 10

³ Oppgave 3: Minimumsheap

I denne oppgaven skal vi bruke en minimumsheap og se hvordan den kan brukes til sortering

- a. Hva er en minimumsheap, og hvilke krav stilles for at det skal kunne kalles en minimumsheap?
- b. Start med minimumsheapen i vedlegget
 - i. Forklar hvordan man legger inn ett tall i en minimumsheap.
 - ii. Legg inn tallet 1 og lag en tegning av svaret ditt for hvert steg i algoritmen.
- c. Start med minimumsheapen i vedlegget
 - i. Forklar hvordan man tar ut ett tall av en minimumsheap.
 - ii. Ta ut ett tall og lag en tegning av svaret ditt for hvert steg i algoritmen.
- d. Start med minimumsheapen i vedlegget
 - i. Forklar hvordan en minimumsheap kan lagres i et array.
 - ii. Skriv opp minimumsheapen i vedlegget som et array.

Skriv ditt svar her...

Maks poeng: 10

4 Oppgave 4: Dobbelt lenket liste

I denne oppgaven skal vi operere med en dobbelt lenket liste (se vedlegget).
I denne oppgaven er følgende viktig:

- Kildekoden skal være kort, oversiktlig, og lett leselig.
 - Skriv funksjonen så effektiv som mulig.
 - Du trenger ikke ta hensyn til spesialtilfeller og indeksskontroll, dvs du kan anta at noden du skal fjerne finnes i listen, og at du ikke skal fjerne første eller siste node.
-
1. Kopier funksjonen `void remove(int index)`, og skriv innholdet i funksjonen der det er markert. Funksjonen skal fjerne noden på plass `index`.
 2. Kopier funksjonen `void remove(char value)`, og skriv innholdet i funksjonen der det er markert. Funksjonen skal fjerne den første noden som har verdi «`value`».

Kopier inn funksjonene fra vedlegget. Husk å skrive ryddig kode.

Maks poeng: 10

⁵ Oppgave 5: Binært søketre

I denne oppgaven skal du lage et balansert binært søketre

- Hva er et balansert binært søketre?
- Ta utgangspunkt i det rød-sortetreet i vedlegget, og legg inn tallet 8.
 - Tegn hvert skritt i algoritmen for rød-sortetree når du legger inn tallet, og beskriv med ord hva du gjør.
- Tegn B-treet av orden 4 som tilsvarer treeet i vedlegget.

Skriv ditt svar her...

Maks poeng: 10

Question 1

Attached



Vedlegg 1: Kildekode

```
public static void oppgave1() {  
    /**  
     * Del 1: Verdier vi skal legge inn  
     */  
    char[] values_1 = "ALFABET".toCharArray();  
    char[] values_2 = "FISK".toCharArray();  
  
    /**  
     * Del 2: Queue her  
     */  
    Queue queue = new LinkedList<Character>();  
    for (char value : values_1) {  
        queue.add(value);  
    }  
  
    System.out.print("Queue first: " + queue.remove());  
    for (int i=0; i<4; ++i) {  
        System.out.print(", " + queue.remove());  
    }  
    System.out.println();  
  
    for (char value : values_2) {  
        queue.add(value);  
    }  
  
    System.out.print("Queue second: " + queue.remove());  
    while (!queue.isEmpty()) {  
        System.out.print(", " + queue.remove());  
    }  
    System.out.println();  
  
    /**  
     * Del 3: Stack her  
     */  
    Stack stack = new Stack<Character>();  
    for (char value : values_1) {  
        stack.push(value);  
    }  
  
    System.out.print("Stack first: " + stack.pop());  
    for (int i=0; i<4; ++i) {  
        System.out.print(", " + stack.pop());  
    }  
    System.out.println();  
  
    for (char value : values_2) {  
        stack.push(value);  
    }  
  
    System.out.print("Stack second: " + stack.pop());  
    while (!stack.isEmpty()) {  
        System.out.print(", " + stack.pop());  
    }  
    System.out.println();  
  
    /**  
     * Del 4: PriorityQueue her  
     */  
}
```



```
Queue pri_queue = new PriorityQueue();
for (char value : values_1) {
    pri_queue.add(value);
}

System.out.print("Priority queue first: " + pri_queue.remove());
for (int i=0; i<4; ++i) {
    System.out.print(", " + pri_queue.remove());
}
System.out.println();

for (char value : values_2) {
    pri_queue.add(value);
}

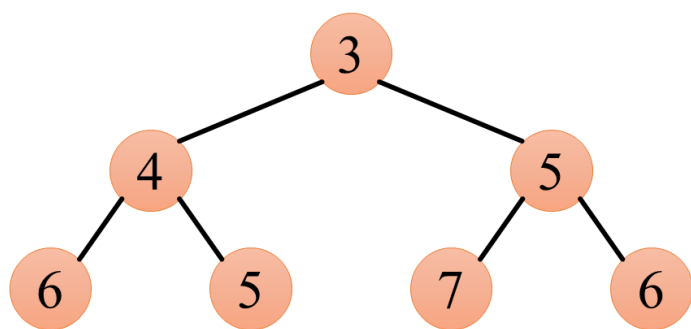
System.out.print("Priority queue second: " + pri_queue.remove());
while (!pri_queue.isEmpty()) {
    System.out.print(", " + pri_queue.remove());
}
System.out.println();
}
```

Question 3

Attached



Vedlegg 3: Heapen



Question 4

Attached



Vedlegg 4: Kildekode

```
public static class DoubleLinkedList {
    public class Node {
        Node next;
        Node prev;
        char value;
    }

    Node head;
    Node tail;

    void addLast(char value) {...}
    void addFirst(char value) {...}
    char removeLast() {...}
    char removeFirst() {...}
    void print() {...}

    void remove(int index) {
        if (index == 0) {
            removeFirst();
        }
        else if (index == size-1) {
            removeLast();
        }
        else {
            // Din kode her
        }
    }

    void remove(char value) {
        // Din kode her
    }
}
```

Question 5

Attached



Vedlegg 5: Rød-sort tre

